IT 314 – Lab 8 (Group submission)

Group 1

code tested: GET method of employeeDepartment API, GET method of employeeProfile API and POST method in createEmployee API

Employee profile:

```
router.get('/', async function(req, res){    // List of all Employee Profiles
    try{
        const employee = await Employee.find({});
        res.send(employee) ;
    }
    catch(err){
        res.json(err)
    }
});
```

Employee department:

```
router.get('/department', async (req,res)=>{
    try{
        res.send(Department)
    }catch(err){
        res.status(500).send(err.message)
    }
}) ;
```

Create new employee:

```
router.post('/', async (req,res)=>{     // Create New employee
    try{

        const data = req.body;

        // to increment no. of employees which will define employee ID
        // only 1 doc in collection
        var query =  await Stat.findOneAndUpdate({},
            {$inc:{NoOfEmployee:1}},
            {new:true}
        );

        await Stat.findOneAndUpdate({},
            {$push: {
            Attendance: {
                key: query.NoOfEmployee,
                value: 0
            }}
        })


        data.EmployeeID = query.NoOfEmployee

        // remove password
        const clone = (({ password, ...rest }) => rest)(data);

        // const result = await Employee.create(newemployee);

        const employee = await new Employee(clone);
        const newEmployee = await Employee.register(employee, data.password);
```

These chunks of codes are tested using Mocha.js and Chai Assert library. For backend in Node.js

Test Cases:

```
const chai = require('chai');
const expect = chai.expect;
const chaiHttp = require('chai-http');
const app = require('../app'); // Your backend app file

chai.use(chaiHttp);

describe('http://localhost:8000', () => {
  describe('GET /employeeDepartment/2', () => {
    it('should return a 200 status code and a JSON object', async () => {
      const res = await chai.request(app).get('/employeeDepartment/2');
      expect(res).to.have.status(200);
      expect(res.body).to.be.an('object');
    });

    it('should return the expected response', async () => {
      const res = await chai.request(app).get('/employeeDepartment/2');
      expect(res.body).to.deep.equal({"_id":"6401ebb4c534c2403966232e","Depart
mentID":1,"DepartmentName":"Backend","NumberOfEmployee":40,"__v":0});
    });
  });

  describe('GET /employeeProfile/2', () => {
    it('should return a 200 status code and a JSON object', async () => {
      const res = await chai.request(app).get('/employeeProfile/2');
      expect(res).to.have.status(200);
      expect(res.body).to.be.an('object');
    });

    it('should return the expected response', async () => {
      const res = await chai.request(app).get('/employeeProfile/2');
      expect(res.body).to.deep.equal({"DateOfJoining":"2023-04-
19T17:18:41.340Z","_id":"6401ec1612eb224fb6fa39b0","EmployeeID":2,"EmployeeNam
e":"KrupalShah","Gender":"Male","Address":"11,
I","PhoneNo":"90","Department":1,"Post":"Empl","Salary":1213,"__v":0});
    });
  });




  describe('POST /createEmployee', () => {
```

```
    it('should return a 200 status code and a JSON object', async () => {
      const res = await chai.request(app)
        .post('/createEmployee')
        .send({ EmployeeID:'1000', EmployeeName:'test', PhoneNo:'1000',
Post:'test', Salary:'1000'});
      expect(res).to.have.status(200);
      expect(res.body).to.be.an('object');
    });

    it('should return the expected response', async () => {
      const res = await chai.request(app)
        .post('/createEmployee')
        .send({ EmployeeID:'1000', EmployeeName:'test', PhoneNo:'1000',
Post:'test', Salary:'1000'});
      expect(res.body).to.deep.equal({ success: true });
    });
  });


});
```

Output of test cases:

```
  http://localhost:8000
    GET /employeeDepartment/2
Database connection successful
Backend
      ✓ should return a 200 status code and a JSON object (66ms)
Backend
      ✓ should return the expected response
    GET /employeeProfile/2
      ✓ should return a 200 status code and a JSON object
      2) should return the expected response
    POST /createEmployee
      ✓ should return a 200 status code and a JSON object (76ms)
      ✓ should return the expected response
```